White Paper Report

Report ID: 109227

Application Number: HD-51674-13

Project Director: Bethany Nowviskie (bethany@virginia.edu)

Institution: University of Virginia

Reporting Period: 5/1/2013-4/30/2014

Report Due: 7/31/2014

Date Submitted: 7/26/2014

National Endowment for the Humanities
Final Report and White Paper
Grant # HD-51674

Speaking in Code

an NEH Summit on the Social and Intellectual Implications
of Tacit Knowledge Exchange in Digital Humanities Software Development

submitted on behalf of the University of Virginia Library Scholars' Lab
by Bethany Nowviskie, MA Ed., Ph.D.
Principal Investigator

grant period: 5/1/2013 – 07/31/2014

IN November 2013, with the generous support of the National Endowment for the Humanities and the University of Virginia Library, the UVa-based Scholars' Lab hosted a summit for digital humanities software developers under the banner of *Speaking in Code*. Thirty-one highly-skilled humanities programmers, designers, and systems architects assembled for two days in Charlottesville, Virginia for *#codespeak* (as it came to be called in social media)—a gathering we believe to be the first extended conversation on the social and intellectual implications of tacit knowledge exchange in DH software development. Importantly, this was a conversation conducted *by advanced practitioners* and *on their own terms.*

Speaking in Code was also unusual in the context of high-level gatherings of US-based technologists, in that 45% of its participants were women and/or people of color. Furthermore, an un-enumerated but anecdotally large percentage of participants volunteered that they spoke to the concerns of the summit from the perspectives of LGBT developers, or as members of some other group typically under-represented at tech conferences and in the ranks of academic software developers. This report: outlines the goals of the summit, its context, and agenda; discusses measures we undertook to welcome a diverse group of participants; sketches themes that emerged in our conversation; and describes outcomes, including steps the Scholars' Lab has taken to promote continued and broadened conversations stemming from *Speaking in Code*.

Context and Conversations at *Speaking in Code*

The call for a more deeply-theorized and critically-engaged digital humanities comes in waves, as regular as the tide, as new scholarly voices join the growing and interdisciplinary chorus of our global DH community. For two groups of people—whose hard-won intellectual understandings, familiarity with humanities questions and methods, and concrete experience with digital project development *should* make them indispensable partners and mentors for scholars new to DH—this call can become as easy to ignore as the roar and retreat of water on sand.

The first of these are longtime practitioners of humanities computing, who worked through a period in which answering almost any digital research question required both pragmatic and theoretical work: scholarly content modeling or database design, from-scratch digitization and a rationalization of structured markup, and the considered, hands-on building of software tools for humanistic inquiry or social platforms for sharing results. To them, the appeal for a theory-aware digital humanities can *itself* sound under-researched—or even (on a bad day) insulting. Joining them are DH software developers—some highly-experienced and others relatively new to the field—whose day-to-day working lives, as heads-down contributors to deadline-driven projects, can underscore the "more hack; less yack" stereotype of THATCamp gatherings.[1] Serious differences in vocabulary, discourse norms, communications platforms, and patterns of work exist between humanities-trained scholars and DH software developers—even those with deep backgrounds in humanistic research.

And yet students and researchers, technologists, cultural heritage workers, and other humanities enthusiasts and supporters operate today under new conditions, with new opportunities. We exploit born-digital contemporary data and digitized historical corpora to ask hitherto unanswerable questions; we augment our time-honored critical and interpretive practices with custom-built tools or cast new light on them through with methods adapted from computationally-intensive disciplines; and we experiment with novel approaches to research, teaching, publication, production, and conservation. For scholars, this work requires increasing technical literacy, a newfound facility in collaborating with the designers and developers of the software platforms that underlie (and can enable or circumscribe) humanities research, and a considerable leap or two of faith. Training programs like the NEH's Institutes for Advanced Technology in the Digital Humanities, and meetings—like *Speaking in Code*—conducted with the support of NEH-ODH Digital Humanities Start-Up Grants are helping to bridge what might otherwise seem a daunting gap.

---

[1] On the appropriation of this phrase for use in strawman arguments, see http://nowviskie.org/2014/on-the-origin-of-hack-and-yack/

[2] For a typical caricature, see: http://dhpoco.tumblr.com/post/72594809525/analytical-building-in-the-digitalhumanities

[3] For these and other critiques, see the select bibliography from *Speaking in Code*: https://github.com/scholarslab/codespeakkit/blob/master/bibliography.md

Too often, though, in the constellation of pedagogical and methodological training opportunities that have grown up in the DH community (including IATDH gatherings, Digital Humanities Summer Institutes, THATCamps and other unconferences, Digital Humanities Conference workshops, and the various programs of HASTAC, NITLE, centerNet, the Association for Computers and the Humanities) the onus has been put on traditionally-trained and -employed humanities scholars to become tech-savvy *digital humanists*, without much attention being paid to the intellectual contributions or the continued professional development of people already steeped in humanities computing technology and for whom this work is a primary focus and responsibility. Many of these are digital humanities software developers—the participants in (and target audience for) our *Speaking in Code* program—who have deep training and professional background in scholarly disciplines, but who lack dedicated opportunities, customized to their special situation as tool-building knowledge workers and to our current moment in humanities scholarship, to grow as programming practitioners, to interrogate and articulate the craft of software development, and to build and sustain a thoughtful and engaged academic culture around code-work in the humanities. A Utopian vision of DH would see scholars engaging with digital humanities software developers as peers in mutually intelligible conversation. The difficulties in realizing this vision, however, are substantial, and consequent misunderstandings and missed opportunities are highly unproductive for the humanities.

Speaking in Code organizers, presenters, and participants held that theories and responses to theory, embedded ethical positions, and sophisticated understandings of the relation of method to humanities interpretation underlie digital humanities software development—often invisibly expressed and embodied in functional code and communal practice. They agreed that it is vital to the collaborative, digital future of scholarship that gaps in communication be overcome and the intellectual basis of DH software development be rigorously and openly discussed.

However, we recognized that, to the broader academic community, several factors in digital humanities software development seem to suggest an unscholarly culture of discourse: the vernacular and artisanal forms of knowledge transmission it depends upon; the parodic, industry-based image of the "brogrammer" (a hyper-masculine stereotype); and the tendency of humanities technologists to work within very small, language- or platform-specific collaborative groups, unrepresented in vectors accessible to and valued by traditionally-trained scholars. From outside the academic computing bubble (and occasionally from within it as well), software developers' discourse can seem: mechanistic or uncritical; exclusionary or unaware of problems of gender, race, and privilege; ephemeral in ways that run counter to the "long tail" of humanities scholarship; driven by commerce and tech-for-tech's-sake; or simply and lamentably unintelligible—conducted in languages and venues too far from the scholarly mainstream to be easily translatable or even noticed. For their part, humanities faculty increasingly encounter the digital humanities on their own, away from the resource-rich environments of digital centers, which can offer a welcoming space for interchange and learning through immersion. Likewise, newer or isolated software developers (even those with deep humanities knowledge gained through graduate training and experience in traditional research, teaching, and publication) can find it difficult to correlate their daily practice with larger intellectual trends in the academy.

Senior and mid-career members of the DH developers' community require organizational structures and systems of reward that will motivate them to communicate with scholars and coders alike, to mentor junior developers, to reach colleagues working in the all-too-common one-programmer shop, and to forge new pathways for intellectual exchange. But first, we must acknowledge that no amount of simple geek-speak translation or scholarly awareness-raising can address the tool-building community's vast collective knowledge about the theoretical basis of code-work in the humanities. The reason is simple: among experts working closely together, in a field that embodies its arguments in made systems rather than in academic prose, whole philosophies of software craftsmanship—and, with them, the intellectual and humanistic underpinnings of an interpretive practice—can seem to go unspoken.

Mutual understanding between humanists and the developers of their tools and frameworks is inhibited by the tendency of humanities computing technologists to work within very small, language- or platform-specific collaborative groups, and to communicate through vectors inaccessible to or under-valued by the broader scholarly community. These include code comments and commit messages, conversations conducted in closed or ephemeral spaces (for instance, within software designed to manage the work of collaborative groups) or through public code snippets that easily lose their context (like GitHub "gists"). We communicate in pull requests, in tickets filed and closed, and in spotty, internal documentation. Software development therefore functions in relative silence within the larger conversation of the humanities, in a sub-culture suffused with tacit understandings and journeyman learning experiences—in many ways, despite its fundamental alphanumeric expression, holding more in common with traditional arts-and-crafts practice than with academic discourse. It is beginning to be understood as a non-discursive hermeneutic of humanities making, still generally unintelligible to scholars who have recently joined the ranks of digital humanists.[2]

The result is felt by the broader academic community in which this work plays a part: reviews of DH projects focus primarily on their content and aesthetic design, and rarely critique underlying tools, user experience models, and computational or software-development approaches in a deep way, indicating appreciation or even basic consciousness of the rich intellectual history and environment from which they stem. And after years of de-valuation as "service providers" within the academy, many small, established, intimate (or perhaps insular) intellectual communities of humanities computing practice now confront the sudden popularity and ubiquity of a fuzzy "digital humanities" label, and the inevitable backlash against it.

The humanities as a whole suffers when DH developers are (or seem, either because of differences in communication modes and styles or because of academic class

[2] For a typical caricature, see: http://dhpoco.tumblr.com/post/72594809525/analytical-building-in-the-digitalhumanities

structures and the strictures of their working lives as support staff members) unprepared to articulate the intellectual basis of digital work and enter into productive conversation about its connection to problems of race, gender, and privilege, or to broader theoretical concerns in humanities scholarship. With concern and goodwill, *Speaking in Code* participants have recently observed—and been generally ill-equipped to engage with—movements like *#transformDH*, which call for critique of so-called digital humanities "code culture" from feminist and other subaltern positions. Even scholars intimately familiar with the design and construction of digital objects and archives, like Alan Liu, are prompted by our silence to ask, "Where is cultural criticism in the digital humanities?"—while heads of major academic programs in humanities computing, like Andrew Prescott of King's College London, observe "a digital humanities community that seems… excessively inward-looking, over-pleased with itself, and lacking in links to wider humanities scholarship."[3]

One goal of the *Speaking in Code* summit was to motivate software developers who have been working quietly, at the heart of the scholarly community, to contribute more visibly to mainstream academic DH discourse from their valuable vantage points and with respect for the new hermeneutic and modes of scholarly production their work represents. In the longer term, we hope the summit and programs like it will provoke contributions on the order of Tara McPherson's examination of the mid-century development of the UNIX operating system to answer the question,

[3] For these and other critiques, see the select bibliography from *Speaking in Code*: https://github.com/scholarslab/codespeakkit/blob/master/bibliography.md

"Why are the digital humanities so white?"—and productive and grounded responses to challenges like Jamie Skye Bianco's self-described "rant against the wielding of code as instrumental, socially neutral or benevolent, and theoretically and politically transparent" (to choose two among many relevant essays in the recent University of Minnesota Press volume, *Debates in the Digital Humanities*).

Speaking in Code planners and participants agreed that, if we are not to waste the opportunities of this current moment of intensified exchange between traditionally-trained scholars and the established community of DH practice, humanities software developers must be better prepared to do three things: to understand the alarming implications of intra- and extramural gaps in scholarly communication; to interrogate both the charges of challengers and their own contributory practices; and to organize themselves both in the shorter and longer term to challenge the new commonplace—voiced alike from within and outside the digital humanities' "big tent"—that DH programmers' ways of working are under-theorized and disconnected from the concerns of the disciplines they serve.

Encouraging Participation ("You Are Welcome Here")

S*peaking in Code* built nicely on a broad survey of our professional landscape conducted by an NEH-funded workshop at the University of Maryland in 2011 (*Off the Tracks*), which focused on "laying new lines for humanities scholars." The work of our summit, however, was more immediately connected to intellectual concerns and community-building in the larger humanities. We aimed to broaden, organize, and invigorate the community of professionals *Off the Tracks* identified and termed "scholar-programmers," by providing them with opportunities to raise the intellectual level of their discourse about code, identify and cultivate leaders and bridge-builders from within the culture, and develop communications vectors through which they may engage in peer mentoring and interrogate their own methods and products.

We took seriously the rare opportunity of our NEH grant to attract deeply-experienced humanities computing developers, and so wanted to indicate that we had not planned conversations suitable for programming novices. And yet we were aware that the barriers to participation in technical conversations are already too high. To that end, we published a statement entitled "You are Welcome Here," and publicized it as part of the Speaking in Code call for participants.

The statement was important for several reasons. First, it expressed the kind of discussion that we wanted to take place. It set the tone for all of the participants, both for developers from traditionally under-represented groups (women, people of color, members of the queer/LGBT community, and others) and for the cisgender white males who make up the largest percentage of software developers inside and beyond DH, and whom we accurately predicted would be most likely to apply. But the statement went beyond a guarantee that we would make our event a safe space. It was an invitation. We wanted to clarify that the organizing team was actively seeking out voices that can be absent from other software development settings—

and to attract participants eager to listen and to hear what their often-marginalized colleagues had to say. The original statement is printed in full below:

> *You Are Welcome Here.*
> This is a small planning and problem-scoping meeting, meant to provide an opportunity for advanced software developers to address shared issues in DH and tacit knowledge—on their own terms. Participants are therefore expected to have contributed to digital humanities software projects at an intermediate or expert level, committing code or with responsibility for aspects of design, architecture, and technical project management that indicates past experience can be brought to bear. However, because software development—even in the more intellectually diverse and welcoming digital humanities—is a predominantly white and male profession, we are particularly committed to amplifying the voices of developers who are women, people of color, queer/LGBT, or otherwise under-represented among programmers, and to creating a friendly and respectful environment for collaboration at this event. Don't let impostor syndrome stop you from applying to Speaking in Code! If you have questions about your application, would like to nominate other participants, or have advice to share, please get in touch with us directly.

We also provided an email address for the entire organizing group and a way to contact the event's primary organizer, a woman, privately. Both options were used.

Applicants submitted simple information via a Google form. This included name, title or role, contact info, current projects or institution, whether they were requesting travel funding, and some brief prose on what they felt they would bring to the discussion and what they might hope to learn. The form was prefaced with this statement:

> Speaking in Code participants will be selected on the basis of their demonstrated experience in digital humanities software development, their interest in advancing solutions to the problems raised by the summit, and the disciplinary and cultural diversity they bring to the conversation. We

> particularly encourage and will prioritize applications by women developers, people of color, LGBT developers, and coders from other under-represented groups.

The statement led several participants to volunteer information that helped our selection committee create a highly diverse group, and we heard privately from many potential applicants, and later from attendees, how important and constructive they found this gesture. It also sparked both praise and criticism in public exchange, samples of which are linked to, below:

https://twitter.com/lwillm/status/365506192644128768
https://twitter.com/Literature_Geek/status/366238036557692930
https://twitter.com/rosy1280/status/365957576581586944
https://twitter.com/veek/status/365815146314076160
https://twitter.com/highermath/status/365887452642148353
http://nataliacecire.blogspot.com/2013/08/tacit.html
https://twitter.com/roopikarisam/status/366300228850171905
http://elotroalex.webfactional.com/recognitions/
http://nowviskie.org/2013/switching-codes/

Nonetheless, the outcome of the call was excellent. We received 42 applications for "Speaking in Code" (not including local organizers and invited presenters), which represented 32 more people than we had requested NEH travel funding to support—and were ultimately able to bring in 32 people to the summit in all (including 22 outside attendees when we had initially planned and budgeted for 10). We did this by working hard to stretch our catering budget, asking applicants who wished to apply for a travel and lodging stipend to research their options and make a specific request, and also by gearing local Scholars' Lab funds toward any overage for the event.[4] Approximately 30% of applicants mentioned in their first contact with us that they wrote from the perspective of an under-represented group, even though that information was not specifically requested as part of the application process.

At the Summit and Afterwards

We began from the premise that humanities software development holds more in common with traditional arts-and-crafts practice than with academic discourse, and that its methods and assumptions may embed a valuable, largely non-verbal new hermeneutic of 'making.' The Speaking in Code program called on four respected, versatile, and highly motivated experts in the field (William J. Turkel, Stéfan Sinclair, Mia Ridge, and Hugh Cayless) to help participants articulate—from their own vantage points—whether and to what degree tacit humanities knowledge is embedded in developers' craft, and to outline the new scholarly and methodological understandings that may be emergent from their praxis. Our agenda was designed to: 1) help participants voice what is particular to the humanities, and intellectually significant for scholarship, in digital humanities software development; and 2) create a space in which they could, in small groups, begin to devise an action-oriented agenda for bridging the gap in critical vocabulary

[4] Ultimately—and despite the expanded guest list—due to lower-than-expected local costs, the government shutdown resulting in a cancellation of the 2013 NEH Project Directors' meeting, and the absence of one budgeted-for speaker, we ended the program with surplus funds, which were applied (with NEH permission) to travel costs associated with publicizing the outcomes from *Speaking in Code* in a peer-reviewed presentation at *Digital Humanities* 2014.

and discourse norms that exists between the developers of humanities platforms and tools, and the scholars who depend upon and critique them.

In the first day's plenary sessions, which were punctuated by discussions and activities, Nowviskie framed the goals of the meeting and led the group in a discussion of summit agreements (including the group decision to operate using a modified version of Chatham House rules). Then, Turkel spoke on tacit knowledge in software development, Sinclair led a design exercise meant to elicit assumptions about commonalities and provoke procedural thinking about the summit community itself, Cayless spoke on the design of text models for language, and Ridge described ways that cultural heritage experts and software developers alike grapple with ambiguity and messiness in data. The day ended with an informal, working dinner—and limerick contest, won by CUNY PhD candidate and developer Micki Kaufman for the following piece of immortal verse:

> The secret to DH-as-racket
> Is to download some source code and hack it.
> Do whatever it takes
> To make sure nothing breaks…
> And never forget that [close bracket]![5]

[5] Kaufman's observations on Speaking in Code are posted online: http://www.mickikaufman.com/2013/12/05/speaking-of-speaking-in-code/ and http://www.mickikaufman.com/2013/12/05/speaking-of-speaking-in-code/ For further samples of participant reflections, see John Laudun's musings: http://johnlaudun.org/20131105-the-theory-behind-it-all/ and https://gist.github.com/johnlaudun/8609385 —and Scott Bailey's notes: https://gist.github.com/csbailey5t/7376635

The second day of the meeting was more loosely structured. Participants broke into small groups to address challenges and opportunities they had identified in the previous day's plenary sessions and at the working dinner.[6]

During the second day, self-selected groups of developers worked on concepts like an annual DH Bug Mash, a user-contributed Zotero library on *Speaking in Code* concerns, and a method for migrating the Association for Computers and the Humanities "DH Answers" platform to a more user-friendly system. Others explored approaches to creating "Humanist-Readable Documentation" and best practices for DH software projects, created plans for new mentoring programs and for a "safe space" virtual community for sharing histories and perspectives, and began a collection of "How to get Started" biographical sketches to encourage novice humanities software developers. Several of these projects are ongoing, under the initiative of Speaking in Code participants. Open and closed-group Google Docs, common code repositories, blog posts, and GitHub gists were shared as part of the process. Participants tweeted throughout the event using the #codespeak hashtag.

To a small extent, by opening critical conversations and strengthening individual ties among often-isolated developers, we believe Speaking in Code improved our overall capacity in the digital humanities to peer-review software and understand the degree to which custom-built algorithms and code are legible expressions of

[6] The full agenda for the *Speaking in Code* meeting is available online: http://codespeak.scholarslab.org/#agenda and annotated here: https://github.com/scholarslab/codespeakkit/blob/master/schedule.md. The only change to the program from plans at the time of our grant submission was that our final speaker, Stephen Ramsay, was unable to attend due to illness. We replaced his keynote talk with plenary discussion on next steps.

theoretical and interpretive stances toward the humanities content they act upon. In some small group sessions at *Speaking in Code*, participants shared and openly critique functional code in the way that so many NEH humanities institutes offer venues for scholars to workshop their readings of works of literature or their interpretations of history and culture. Like our colleagues in the emerging area of Critical Code Studies, the practicing DH developers who came together in Charlottesville agreed that there is a poetics to code—but their unique and fundamental stance was to treat code not as poetry to be analyzed but as *poiesis*, as a making to be reasoned, voiced, emulated, and progressively, collaboratively refined. We read code together, not only as a text or a system of social and cultural signifiers, but as the considered orchestration of specific tasks and goals relevant to humanities work, and by extension to practitioners of humanities scholarship. More importantly, in small groups and in plenary sessions, we offered readings of our own, seldom-discussed practices, workflows, and embedded assumptions—and thereby came to understand them better.

All this said, the staff of the Scholars' Lab will only consider *Speaking in Code* a true success to the degree that it lights a crucial spark under the digital humanities developers' community in which we participate, a community whose intellectual work both drives and responds to humanities scholarship in the 21st century. NEH support helped us begin creating a social framework through which humanities software developers could find their voices in three important ways: in clarifying and communicating to a wider audience the deeply theoretical and philosophical basis of code-level contributions to humanities scholarship; in engaging more consciously in the mentorship and hybrid scholarly and professional development of their junior colleagues (particularly colleagues from under-represented groups); and in seeding critical, participant-driven initiatives we could not predetermine. It is our hope that *Speaking in Code* participants will use the experience of the summit to make a positive impact not only on the status and sophistication of technical work in the digital humanities, but on the larger shape of collaborative humanities research and new-model scholarly communication in the digital age. We also hope to see further public conversations about the summit's outcomes and themes.

To that end, in late July 2014, we released a "*Speaking in Code* kit," in order to make it as easy as possible for other groups to host face-to-face events like ours, and to encourage interested individuals to open up new exchanges online. The kit contains our starter bibliography, several pages of advice for welcoming a diverse group of participants and for planning an extended gathering, and the framework of a Jekyll website (complete with instructions), ready to be forked in Git or scaffolded using Yeoman in Node, and published at no cost on GitHub Pages. The kit is available at https://github.com/scholarslab/codespeakkit.

We also created an IRC channel on Freenode (#speakingincode) and have further promoted the hashtag developed for the November meeting in Charlottesville: #codespeak, for general discussion and for use by future *Speaking in Code* hosts. Fourteen contributions to our DH Developer "Origin Stories" collection—an idea hatched at the November meeting—have been published at http://codespeak.scholarslab.org/starting/. They address the question, "How do I get started as a DH software developer?" (or interrogate the question, or modify it in helpful ways). Summit participants and interested developers who could not attend are invited to contribute to the collection by submitting a pull request on the Speaking in Code GitHub repository or submitting a Markdown file to the Scholars' Lab.

Finally, we updated the original event webpage (http://codespeak.scholarslab.org/) to contribute to the "kit" and serve as a lasting record of the NEH summit. We announced these developments on Twitter, on the Scholars' Lab website and Facebook page, at CLIR's Re:Thinking blog community and on individual participants' and organizers blogs, and have plans to continue bringing them to the attention of the community in a variety of venues as the new academic year begins. Uptake and interest in the first days after release has been strong.

We are grateful to the National Endowment for the Humanities and the University of Virginia Library for supporting this meeting, to colleagues from other institutions who formed our planning group, to Scholars' Lab co-workers for their assistance at the event, and most of all to our inaugural *Speaking in Code* participants—who contributed to conversations many of us found remarkably frank and heartfelt, intellectually stimulating, and deeply restorative and promising of good work to come.